A Boosting Ant-Colony Optimization Algorithm for Computer Intrusion Detection

Emad Soroush, Mohammad Saniee Abadeh, and Jafar Habibi

Abstract—**This paper proposes a boosting Ant-colony optimization algorithm for intrusion detection in computer networks. The goal of the algorithm is to extract a set of classification rules from a network dataset. These rules are capable of detecting normal and abnormal behaviors. The presented algorithm is evaluated according to some measures like detection, false alarm, and classification rates. Results show that the proposed boosting algorithm is capable of producing a reliable intrusion detection system.**

I. Introduction

NOWADAYS, security of the information in the computer networks has been one of the most important points of discussion. To preserves the secure condition it is essential to be aware of the behavior of the incoming data. Is it a normal or abnormal data? It is a too vulnerable and complicated Question. Owing to the fact that intrusive data are in several and similar forms, distinguishing them from the normal ones is so astounding. In other words, the boundaries of the normal and abnormal behaviors are not well defined. The problem of intrusion detection has been studied extensively in computer security [1]-[4], and has received a lot of attention in machine learning and data mining [5]-[7]. Fundamentally, two models of intrusion detection have been introduced [8]: Anomaly Detection: This model first constructs the normal profile that contains metrics extracted from the system operation. While monitoring the system, current observation is compared with the normal profile in order to detect changes in the patterns of utilization or behavior of the system. Another one is the Signature or Misuse Detection: This technique relies on the patterns of the known intrusions in order to match and identify intrusion. In this case, the intrusion detection can be viewed as a classification problem. Ant-colony optimization algorithm is an evolutionary learning algorithm which could be applied to solve the combinatorial optimization problems [9]-[11]. ACO algorithm fundamental idea has been inspired by the behavior of the real ants. Ants deposit pheromone as a trace to direct the other ones in finding foods. They choose their path according to the congestion of the pheromone. The above behavior of the real ants has inspired an algorithm which a set of artificial ants, as a group of simple agents, cooperate with each other to solve a problem by exchanging information via pheromone deposited on the graphs of the edges. Pheromone acts like a distributed memory for communicating ants with each other. This algorithm creates an ant system applied to many combinatorial optimization problems such as traveling salesman problem (TSP) [12]-[15] and the quadratic assignment problem [16], [17]. Furthermore, ACO has been used in the context of data mining for clustering [18], and classification [19] problems.

Ant-Miner [19] is an ant colony based system which is used for the classification task of data mining. In this algorithm, the objective is to assign each case (record, or instance) to one class, from the set of predefined classes, based on the values of antecedent attributes (called predictor attributes) of the case. Discovered knowledge gained from the classification task is shown by a set of rules. Each rule consists of two parts.

$$IF < term1 \,\&\, term2 \,\&\, ... > THEN < class > \tag{1}$$

IF part includes of some terms (predicator attribute) which are connected by the logical operation "and (&)". the organization of each term is a triple like

$$< attribute \;, operator \;, value \; > \tag{2}$$

Such as : < Sex = male>. THEN part indicates the predicated class for the cases which their attributes satisfy all the terms in the antecedent rule part.

Results of Ant-Miner on some of the domains show that ACO has an intense ability to advance in the classification task field of data mining. The details of Ant-Miner would be talked later.

This paper is organized as follow: first reviewing of Ant-Miner algorithm and its superiority in comparison to the other ones. In the Next section, it will be introduced the variations which produce the boosting Ant-Miner. Then in the next section, experimental results will be analyzed and finally, draw some conclusions.

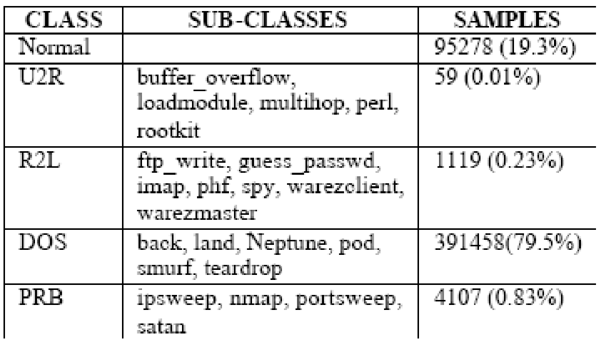
II. Intrusion Dataset

In the 1998 DARPA [20], intrusion detection evaluation programme, an environment was set up to get raw TCP/IP dump data for a network by simulating a typical US Air Force LAN. The LAN was operated like a real environment, but was blasted with several attacks. For each TCP/IP connection, 41 various quantitative and qualitative features were extracted. This dataset contains 494,021 patterns. Fig 1 represents some details about the different classes of this dataset.

Each of the attack classes of intrusion detection problem is discussed as follow: [21]

Manuscript received September 18, 2005.

All of the Authors are with the Department of Computer Engineering at Sharif University of Technology (phone: +98-9133117532; fax: +98-66019246; e-mail: saniee@ce.sharif.edu).



CLASS	SUB-CLASSES	SAMPLES
Normal		95278 (19.3%)
U2R	buffer_overflow, loadmodule, multihop, perl, rootkit	59 (0.01%)
R2L	ftp_write, guess_passwd, imap, phf, spy, warezclient, warezmaster	1119 (0.23%)
DOS	back, land, Neptune, pod, smurf, teardrop	391458(79.5%)
PRB	ipsweep, nmap, portsweep, satan	4107 (0.83%)

Fig 1 : class description of intrusion detection dataset

1. Probing (PRB)

Probing is a class of attacks where an attacker scans a network to gather information or find known vulnerabilities. An attacker with a map of machines and services that are available on a network can use the information to look for exploits. There are different types of probes: some of them abuse the computer's legitimate features; some of them use social engineering techniques. This class of attacks is the most commonly heard and requires very little technical expertise.

2. Denial of service (DOS)

DOS is a class of attacks where an attacker makes some computing or memory resource too busy or too full to handle legitimate requests, thus denying legitimate users access to a machine There are different ways to launch DOS attacks: by abusing the computer's legitimate features; by targeting the implementations bugs; or by exploiting the system's misconfigurations. DOS attacks are classified based on the services that an attacker renders unavailable to legitimate users.

3. User to root (U2R)

User to root exploits are a class of attacks where an attacker starts out with access to a normal user account on the system and is able to exploit vulnerability to gain root access to the system. Most common exploits in this class of attacks are regular

4. Remote to user (R2L)

A remote to user attack is a class of attacks where an attacker sends packets to a machine over a network, then exploits machine's vulnerability to illegally gain local access as a user. There are different types of R2U attacks: the most common attack in this class is done using social engineering.

III. RELATED WORKS

What is intrusion detection? A well-known definition for it is as follow: Detecting unauthorized use, misuse and attacks on information systems is defined as intrusion detection [22]. As a result of so many experiments, the most appropriate method to identify the intrusions is scrutinizing the logged data generated by the operating system and network. It is also useful to gather the data which has been stored even after intrusion detection process. These valuable records would be profoundly analyzed, and it could be possible to extract the patterns of different intrusions for prevention of the future attacks. The Intrusion Detection System (IDS) is not only an analytical tool, but also is a detective and predictive one. Soft computing techniques are being extensively used by the IDS community because of their generalization characteristics which help in detecting known and unknown intrusions which was not described by any pattern. Earlier researches on IDS tend to use of the rule-based approach. But it has a difficulty in detecting new attacks or attacks that had no previously described patterns [23]-[26]. Lately, the emphasis is being shifted to learning by examples and data mining paradigms. Neural networks have been extensively used to detect both misuse and anomalous patterns [27]-[31]. Recently, kernel-based methods, SVMs and their variants are being used to detect intrusions. Many researchers used data mining techniques to identify key patterns that help in detecting intrusions [32]-[34]. Distributed agent technology is being suggested by a few researchers to overcome the inherent limitations of the client–server paradigm and to detect intrusions in real [35]-[38].

IV. DATA MINING USING ACO

As it was mentioned before, ACO algorithm has recently been used in various kinds of data mining problems such as clustering, and classification. Ant-Miner algorithm is the result of applying ACO for discovering rules [19]. This learning algorithm utilizes of the artificial simple ants in order to explore the training search space and gradually make candidate rules. Each tour of the ants is directed by the combination of pheromone and an Entropy function. Then among the candidate rules, the best one is selected and augmented to the discovered rules. This process will be done iteratively and finally a set of rules would be discovered which they could be used as our detection patterns to apply in larger data sets. This algorithm is presented in Fig. 2.

At first the rule list is empty (line 3). Then the algorithm insert into the main loop (line 4). Pheromone is initialized and then algorithm inserts into the inner loop (line 7). In this part each ant gradually makes a rule (line 8). As it was mentioned, each term in the antecedent part of the rule is organized in the following pattern:

$$< attribute = value > \qquad (3)$$

Then the selection of a value for each attribute is performed according to a hybrid function which is a combination of ant's pheromone and Entropy heuristic function (line 9). After generating a new rule, pruning function will be called (finding simpler rule with the higher quality), and after updating the pheromone will be done (line 10 and 11). Once the number of the rules reaches to the No_of_ants threshold or recently rules converge to a specific rule, the constraint of the inner loop will be satisfied (line 17). Then the best quality rule should be selected and added to the discovered rule list (lines 18 and 19). In this step the Training-Set will be updated (line 20). This iteration will be repeated until the Training-Set is greater than the Max_ uncovered _cases.

```
1   A High-Level Description of Ant-Miner
2   Training-Set = {all training cases}
3   DiscoveredRuleList = [ ] /* rule list is empty at first */
4   WHILE (TrainingSet > Max_uncovered_cases)
5   t = 1    /* ant index */
6   j = 1    /* convergence test index */
7   initialize all trails with the same amount of pheromone.
8   REPEAT
9   Ant_t starts with an empty rule and incrementally
    constructs a classification rule R_t by adding one term at a
    time to the current rule
    *********************
10  Prune rule R_t
    *********************
11  Update the pheromone of all trails by increasing
    pheromone in the trail followed by Ant (proportional to
    the quality of R_t ) and decreasing pheromone in the other
    trails (simulating pheromone evaporation)
    *********************
12  IF (R_t is equal to R_t-1) /* update convergence test */
13  THEN j = j + 1
14  ELSE j = 1
15  END IF
16  t = t + 1
17  UNTIL(t>=No_of_ants)OR(j>=No_rules_converge)
18  Choose the best rule R_best among all rules R_t constructed
    by all the ants
19  Add rule Rbest to DiscoveredRuleList
20  Training-Set = Training-Set -{set of cases correctly
    covered by Rbest }
21  END WHILE
```

Fig 2: Ant-Miner algorithm[11]

1.1. Entropy function

For each $A_i = V_{ij}$ (A_i is the *i*th attribute and V_{ij} is the *j*th possible value of this attribute) an Entropy function is computed according to equation (4)

$$H(W \mid A_i = V_{ij}) = -\sum_{w=1}^{k} P(w \mid A_i = V_{ij}).\log_2 P(w \mid A_i = V_{ij})) \quad (4)$$

where W is the class attribute (i.e., the attribute whose domain consists of the classes to be predicted), and *k* is the number of classes. P is the probability of observing class *w* if $A_i = V_{ij}$ has been observed.

In summary the higher, $H(W \mid A_i = V_{ij})$ the more uniformly the class distributed.

1.2. Quality computation

In Ant-Miner the quality of a rule is computed according to equation (5).

$$Q = TP/(TP + FN) * TN/(TN + FP) \quad (5)$$

TP: true positives, the number of cases in our training set covered by the rule that have the class predicted by the rule.
FP: false positives, the number of cases covered by the rule that have a class different from the class predicted by the rule.
FN: false negatives, the number of cases that are not covered by the rule but that have the class predicted by the rule.
FN: true negatives, the number of cases that are not covered by the rule and that do not have the class predicted by the rule. As it is mentioned, the Quality is used for selecting the best rule between discovered ones. Besides, Ant-Miner uses the quality as a factor for pheromone updating.

1.3. Pheromone updating

In each iteration t, the pheromone will be increased for all the terms including in the constructed rule.

$$T_{ij}(t+1) = T_{ij}(t) + T_{ij}(t) * Q, \forall T_{ij} \in R \quad (6)$$

where R is the set of all the terms included in the rule. Furthermore, the pheromone should be decreased for every Tij not in the antecedent part of the rule. By normalizing the pheromone this objective would be satisfied.

1.4. Classification of new Cases

In order to classify a new test case, discovered rules are inserted into a sorted list according to their production orders. The first rule that covers the new record would be the indicator. Then the class predicted by that rule is compared with the class of the case. It is possible that no rule of the list covers the new case. In this situation, the new case is classified by a default rule that simply predicts the majority class in the set of uncovered training cases.

1.5. Thresholds

AntMiner algorithm uses some thresholds in order to satisfy some constraints in both iterations of the algorithm. These thresholds are as follow:
Number of ants (No_of_ants): it indicates the maximum number of the ants involved in rule discovery process.
Minimum number of cases per rule (Min_cases_ per_rule): Each rule must cover at least Min_ cases_ per_rule to ensure a minimum generality.
Maximum number of uncovered cases in the training set (Max_uncovered_cases): it is used as an ending constraint to terminate the main loop.
Number of rules used to test convergence of the ants (No_rules_converge): If the current ant has constructed a rule that is exactly the same as the rules constructed by the No_rules_converge -1 previous ants, then convergence has occurred. Therefore the current iteration of the main loop of the Algorithm is stopped and the next iteration is started.

V. NEW BOOSTING APPROACH

As it is mentioned in the next section, our experiment of Ant-Miner on intrusion data set is resulted a considerable high performance in detection rate and false alarm rate.

Besides, its classification rate is competitive. So our new Approach insists on improving its classification rate. In this variation of Ant-Miner, we attempt to preserve the benefits of Ant-Miner as much as possible. We reach to our Boosting Ant-Miner by applying the following updates:

1. Dataset partitioning

One of the disadvantages of the original Ant-Miner is that it does not assign the same chance to each of the classes in the rule discovery process. To overcome this problem, we have partitioned the main Training-set according to the different classes. The algorithm is then applied to each of these sets separately.

2. Changing the view of Quality

The computation of the quality of each rule is changed by considering the whole Training-set during the algorithm operation. By this variation, we hope to select more general rules. Besides, because each part contains only one class, it is essential in order to cope with the original formulation of the quality, selecting them in basis of the whole Training-set not the separated Training-sets.

3. Heuristic function conversion

After dataset partitioning, there are various candidates for the selection of a heuristic function. If the original Entropy function is used, it will be converted to the following formula:

$$H(W \mid A_i = V_{ij}) = P(w \mid A_i = V_{ij}).\log_2 P(w \mid A_i = V_{ij})) \tag{7}$$

This formula is not as efficient as its original version. For instance it should be considered the H function when P approaches to zero or one. In both conditions H approach to the zero result. But only in one of this condition this result is proper ($p \to 1$). Notice that in the previous definition of H, the formula was a sigma not only a single expression. The Heuristic function is updated according to equation (8).

$$H(W \mid A_i = V_{ij}) =_{ij} Arc\cos(p) \tag{8}$$

In the next section, we will examine the effect of both mentioned equations for the heuristic function.

4. Change the pruning technique

In the last variation, we change the step of applying the pruning procedure. Instead of pruning each rule in the inner iteration, we only prune the best rule which is discovered and inserted into the discovered rules. In the other word, using the suggested pruning update for the algorithm will decrease the running time and increase the performance of the whole algorithm significantly. One of the reasons that the operation of the algorithm improves using the proposed pruning technique is that the pheromone updating process will be done more effectively. However it is necessary to investigate the effect of different pruning techniques to obtain more efficient classification system.

After applying these new variations, the new Ant-Miner algorithm performs for each minor Training-set independently. In each of these datasets we gain a group of rules. Then the first rule will be selected (like the method of original Ant-Miner in classifying the new cases) and insert it into the discovered rules. After performing this operation on all of the separated Training-sets, the total size of the discovered rules is equal to the number of different classes. Then we apply these rules to the whole Training-set and eliminate the cases (records) which covered by these rules. Now the original Ant-Miner is applied on the remained Training-set records. As a result we gain another set of rules and add it to the end of the ordered rule list. So we now have a list of rules and could utilize it in other new datasets. The method of classifying new cases is the same as the original Ant-Miner.

VI. EXPERIMENTAL RESULTS

In the experiments, first some of the variation results of the Ant-Miner have been proposed. Then the original Ant-Miner and its new approach have been compared with the other data mining classification algorithm. Some abbreviation used in the following tables is as follow:

FAR: **F**alse **A**larm **R**ate
DR: **D**etection **R**ate
CRi: **C**lassification **R**ate of the class number **i**

Class Number	Class Name
0	DOS
1	Normal
2	PBR
3	R2L
4	U2R

NR: **N**umber of **R**ules in set
MTR: **M**edian of **T**erms in **R**ules.
Ave: Average
STDEV: Standard Deviation

In Table 1 to 5, the original Ant-Miner and its variations have been analyzed. Each of the new approaches is run ten times.

Table 1: Original Ant-Miner

	CR0	CR1	CR2	CR3	CR4
B-Rule	98.6	78.6	98.4	86.4	93.2
Ave	99.1	67.8	44.4	85.9	37.3
STDEV	0.003	0.285	0.398	0.117	0.405

Table 2: Ant-Miner with change of Quality view (Q-Ant-Miner)

	CR0	CR1	CR2	CR3	CR4
B-Rule	99.0	78.6	37.3	93.6	39.0
Ave	99.3	71.9	31.8	82.7	9.8
STDEV	0.003	0.256	0.117	0.126	0.188

Table 3: Entropy Deletion and change of Quality view (ED-Ant-Miner)

	CR0	CR1	CR2	CR3	CR4
B-Rule	94.5	91.4	95.9	91.3	64.4
Ave	98.2	78.2	43.9	90.7	37.1
STDEV	0.021	0.063	0.262	0.028	0.272

Table 4: Dataset Partitioning with Entropy = p * log p (DPE1-Ant-Miner)

	CR0	CR1	CR2	CR3	CR4

B-Rule	97.5	78.6	94.3	92.9	55.9
Ave	91.9	76.3	66.2	72.7	57.4
STDEV	0.122	0.094	0.288	0.141	0.154

Table 5: Dataset Partitioning with Entropy = ArcCos p (DPE2-Ant-Miner)

	CR0	CR1	CR2	CR3	CR4
B-Rule	98.4	94.0	94.3	92.4	67.8
Ave	97.2	81.5	87.4	86.7	38.3
STDV	0.009	0.142	0.191	0.128	0.262

As it is obvious in Table 1, original Ant-Miner does not have a good result on CR2 and CR4. In Table 2, the only change has been the Quality view of the algorithm (Q-Ant-Miner). It is noticeable that the results do not improve. We get the worse results Especially in CR4 (9.8) and CR2 (31.8). Also in basis of STDEV none of them is superior to another one. In Table 3, also Entropy Heuristic will be eliminated from the original Ant-Miner (ED-Ant-Miner). This new approach could improve the result of Table 2 (Q-Ant-Miner), but the result is not changed significantly in comparison to the original Ant-Miner. Due to the elimination of Entropy, the time of searching time has extremely increased. It is because of Pheromone that is the only factor for constructing rule and it takes time to learn itself (at the start of algorithm we do not have any pheromone). At Table 4 Dataset partitioning is considered. Also change of Quality view and Heuristic Function conversion is augmented. The H function is converted as follow:

$$H(W \mid A_i = V_{ij}) = P(w \mid A_i = V_{ij}).\log_2 P(w \mid A_i = V_{ij})) \quad (9)$$

In this approach (DPE1-Ant-Miner) we greatly improved the CR. CR2 (66.2) and CR4 (57.4) have extremely improved. In addition as it is reported in Table 6, DPE1-Ant-Miner has a considerable improvement in DR and FAR. The only disadvantageous is the reduced CR3 (72.7). In general the results are boosted a lot. In Table 5, The H function is converted as follow:

$$H(W \mid A_i = V_{ij}) =_{ij} Arc\cos(p) \quad (10)$$

Also the new pruning technique would be applied (DPE2-Ant-Miner). The results are much better than the other ones. In all the classification fields this new approach is better or near to others. Only the previous approach (DPE1-Ant-Miner) is superior in CR4. Although detection rate is decreased a bit, but lower FAR is obtained in comparison to the other approaches. In order to understand the magnificence of this result, it is better to compare it with the original Ant-Miner. The performance improvement of the new approach has been revealed in CR2 and CR3. FAR has improved more than one percent. And also the STDEV is competitive in this approach.

In Table 6, a list of classifying algorithms which are applied on intrusion dataset is reported. Also various kinds of Ant-Miner approaches are compared to one another. Apparently, Ant-Miner different approaches have a considerable superiority in comparison to other algorithms. Among various Ant-Miner approaches, DPE1-Ant-Miner and DPE2-Ant-Miner are selected as the best. Both of them are based on dataset partitioning approach. The DPE1-Ant-Miner increases DR significantly. But the DPE2-Ant-Miner has greatly improved FAR. In general, a hybrid of various variations of the proposed Ant-Miner might result a more reliable intrusion detection system. In addition there also some concepts which requires more investigation such as: The effect of pruning variations on Ant-Miner, finding a more appropriate Heuristic Function, and altering the method of updating pheromone.

Table 6: Algorithm comparison

Algorithm	FAR %	DR %
Original Ant-Miner [11]	3.50	99.40
Q-Ant-Miner	3.0	99.1
ED-Ant-Miner	2.5	99.3
DPE1-Ant-Miner	**2.7**	**99.5**
DPE2-Ant-Miner	**2.1**	**99.1**
SRPP[39]	3.85	99.08
NCP [40]	0.66	98.78
EFRID[41]	7	98.95
RIPPER-Artificial Anomalies[42]	2.02	94.26
SMARTSIFTER[43]	0	82

VII. Conclusion

In this paper a boosting Ant Colony Optimization algorithm is presented. The suggested algorithm is used to generate classification rules for intrusion detection problem. The proposed algorithm is an improved version of Ant-Miner which is a rule discovery solution in classification problems. The main new features of the presented algorithm are as follows:

1) The algorithm partitions the original Training-Set to some sub-partitions. Each of these sub-partitions is related to a specific class of the classification problem.

2) The view of quality assignment for each rule is based on the original Training-Set. Note that in the original Ant-Miner, the Training-Set is updated each iteration.

3) New heuristic functions are applied to the boosting Ant-Miner. These heuristics enabled the algorithm to improve the detection rate and false alarm rate independently.

4) The step of applying the pruning procedure is changed from inner iteration to the outer one. This modification caused a significant improvement for the presented algorithm. The effect of this modification is left to our future work.

The performance of the boosting Ant-Miner is compared to some well-known intrusion detection systems. Results show that the presented algorithm has improved the performance of the intrusion detection system considerably.

References

[1] Heady R, Luger G, Maccabe A, Sevilla M. The architecture of a network-level intrusion detection system, Technical report, CS90-20, Department of Computer Science, University of New Mexico, Albuquerque, NM87131.

[2] Amoroso E. Intrusion detection. Intrusion.net Books, January 1999.

[3] Allen J, Christie A, Fithen W, McHugh J, Pickel J, Stoner E. State of the practice of intrusion detection technologies. Technical report CMU/SEI99-TR-028, ESC-99-028, Carnegie Mellon, Software Engineering Institute, Pittsburgh, Pennsylvania; 1999.

[4] Axelsson S. Intrusion detection systems: a survey and taxonomy. Technical report no. 99-15, Department of Computer Engineering, Chalmers University of Technology, Sweden. March 2000.

[5] Sundar J, Garcia-Fernandez J, Isaco D, Spafford E, Zamboni D. An architecture for intrusion detection using autonomous agents, Technical report 98/05, Purdue University, 1998.

[6] Crosbie M. Applying genetic programming to intrusion detection. In: Proceedings of the AAAI 1995 fall symposium series, November 1995.

[7] Lee W, Stolfo SJ, Mok KW. Mining audit data to build intrusion detection models. In: Proceedings of international conference on knowledge discovery and data mining (KDD'98), 1998. p. 66–72.

[8] Axelsson S. Intrusion detection systems: a survey and taxonomy. Technical report no. 99-15, Department of Computer Engineering, Chalmers University of Technology, Sweden. March 2000.

[9] M. Dorigo, A. Colorni, and V. Maniezzo, "The ant system: Optimization by a colony of cooperating agents" IEEE Trans. Syst. Man Cybern. B, vol. 26, pp. 29–41, Feb. 1996.

[10] M. Dorigo and G. Di Caro, "The ant colony optimization meta-heuristic" in New Ideas in Optimization, D. Corne, M. Dorigo, and F. Glover, Eds. New York: McGraw-Hill, 1999, pp. 11–32.

[11] M. Dorigo, G. Di Caro, and L. M. Gambardella, "Ant algorithms for discrete optimization" Artif. Life, vol. 5, no. 2, pp. 137–172, 1999.

[12] A. Colorni, M. Dorigo, and V. Maniezzo, "Distributed optimization by ant colonies" in Proc. ECAL91—Eur. Conf. Artificial Life. New York: Elsevier, 1991, pp. 134–142.

[13] M Dorigo, V Maniezzo, A Colorni "An investigation of some properties of an ant algorithm" in Proc. Parallel Problem Solving from Nature Conference (PPSN 92) New York: Elsevier, 1992, pp. 509–520.

[14] M. Dorigo, "Optimization, learning and natural algorithms" Ph.D. dissertation, DEI, Politecnico di Milano, Italy, 1992 (in Italian).

[15] M. Dorigo, V. Maniezzo, and A. Colorni, "The ant system: Optimizationby a colony of cooperating agents" IEEE Trans. Syst, Man, Cybern. B, vol. 26, no. 2, pp. 29–41, 1996.

[16] V. Maniezzo, A. Colorni, and M. Dorigo, "The ant system applied to the quadratic assignment problem" Universit´e Libre de Bruxelles, Belgium, Tech. Rep. IRIDIA/94-28, 1994.

[17] L. M. Gambardella, E. Taillard, and M. Dorigo, "Ant colonies for QAP" IDSIA, Lugano, Switzerland, Tech. Rep. IDSIA 97-4, 1997.

[18] N. Monmarché, "On data clustering with artificial ants" in Data Mining with Evolutionary Algorithms, Research Directions – Papers from the AAAI Workshop, A. Freitas, Ed. Menlo Park, CA: AAAI Press, 1999, pp. 23–26.

[19] RS Parpinelli, HS Lopes, AA Freitas "Data Mining With an Ant Colony Optimization Algorithm"- IEEE Transactions on Evolutionary Computation, 2002

[20] http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html KDD-cup data set.

[21] Srinivas M, Andrew HS, Ajith A. Intrusion detection using an ensemble of intelligent paradigms. Journal of Network and Computer Applications, 2004, in press, corrected proof, available online 28 February 2004.

[22] Denning D. An intrusion-detection model. IEEE Transactions on Software Engineering 1987;SE- 13(2):222–32.

[23] Anderson D, Lunt TF, Javitz H, Tamaru A, Valdes A. Detecting unusual program behavior using the stastistical component of the next-generation intrusion detection expert system (NIDES). SRI-CSL-95-06. Menlo Park, CA: SRI International; 1995.

[24] Ilgun K. USTAT: a real-time intrusion detection system for UNIX. In: Proceedings of the 1993 Computer Society symposium on research in security and privacy, Oakland, CA, May 24–26, Los Alamitos, CA. IEEE Computer Society Press; 1993. p. 16–29.

[25] Lunt T, Tamaru A, Gilham F, Jagannathan R, Jalali C, Neumann PG, Javitz HS, Valdes A, Garvey TD. A real time intrusion detection expert system (IDES)—final report. Menlo Park, CA: SRI International; 1992.

[26] Mukkamala S, Janoski G, Sung AH. Intrusion detection using neural networks and support vector machines. Proceedings of IEEE international joint conference on neural networks, 2002. p. 1702–7.

[27] Cannady J. Artificial neural networks for misuse detection. In: National information systems security conference, 1998. p. 368–81.

[28] Debar H, Dorizzi B. An application of a recurrent network to an intrusion detection system. In: Proceedings of the international joint conference on neural networks, 1992. p. 78–83.

[29] Debar H, Becke B, Siboni D. A neural network component for an intrusion detection system. In: Proceedings of the IEEE Computer Society symposium on research in security and privacy, 1992. p. 240–50.

[30] Mukkamala S, Sung AH. Feature selection for intrusion detection using neural networks and support vector machines. Journal of the Transport Research Board National Academy, Transport Research Record No. 1822 2003:33–9.

[31] Riedmiller M, Braun H. A direct adaptive method for faster back propagation learning: the RPROP algorithm. Proceedings of the IEEE international conference on neural networks, San Francisco, 1993.

[32] Jianxiong L, Bridges SM. Mining fuzzy association rules and fuzzy frequency episodes for intrusion detection. International Journal of Intelligence Systems 2000;15(8):687–704.

[33] Moller AF. A scaled conjugate gradient algorithm for fast supervised learning. Neural Networks 1993(6):525–33.

[34] Steinberg D, Colla PL, Kerry M. MARS user guide. San Diego, CA: Salford Systems; 1999.

[35] Crosbie M, Spafford EH. Defending a computer system using autonomous agents. Technical report CSDTR- 95-022, 1995.

[36] Dasgupta D. Immunity-based intrusion detection system: a general framework. In: Proceedings of 22nd the national information systems security conference (NISSC), 1999. p. 147–60.

[37] Helmer G, Wong J, Honavar V, Miller L. Lightweight agents for intrusion detection. Journal of Systems and Software 2003:109–22.

[38] Porras A, Neumann PG. EMERALD: event monitoring enabling responses to anomalous live disturbances. In: Proceedings of the national information systems security conference, 1997. p. 353–65.

[39] M. Saniee Abadeha,J. Habibia, C. Lucasb,"Intrusion detection using a fuzzy genetics-based learning algorithm",Journal of Network and Computer Applications, In Press, 2005

[40] Ishibuchi H, Nakashima T, Kuroda T. A hybrid fuzzy genetics-based machine learning algorithm: hybridization of Michigan approach and Pittsburgh approach. In: Proceedings of 1999 IEEE international conference on systems, man, and cybernetics, vol. I, October 12–15, Tokyo, Japan, 1999.p. 296–301.

[41] Gomez J, Dasgupta D. Evolving fuzzy classifiers for intrusion detection. In: Proceedings of the 2002 IEEE workshop on information assurance, United States Military Academy, West Point, NY, June 2001.

[42] Fan W, Lee W, Miller M, Stolfo SJ, Chan PK. Using artificial anomalies to detect unknown and know network intrusions. In: Proceedings of the first IEEE international conference on data mining, 2001.

[43] Yamanishi K, Jun-ichi Takeuchi, Williams G. On-line unsupervised outlier detection using finite mixtures with discounting learning algorithms. In: Proceedings of the sixth ACMSIGKDD international conference on knowledge discovery and data mining, 2000. p. 320-4.